AMENDMENTS TO THE CLAIMS

Claims 1-11, 16-20, 26-29, and 35-42 were pending at the time of the Office Action.

Claims 12-15, 21-25, and 30-34 are withdrawn from consideration.

Claim 4 is canceled.

Claims 1, 3, 5-9, 11, 16-19, 26-29, and 35-41 are amended.

Claim 43 is added.

Claims 1-3, 5-11, 16-20, 26-29, and 35-43 remain pending.

1. (Currently Amended) A method for managing access to resources, comprising:

generating a list of resource signatures, each of the resource signatures being generated based at least on function names included in an import table of a corresponding resource;

accessing the ~~a~~ list of resource signatures, each of the resource signatures configured with ~~being classified to indicate~~ an accessibility status, wherein the accessibility status includes one of loadable and restricted ~~of a corresponding resource~~;

generating a verification signature for a requested resource, the verification signature being generated based at least on function names included in an import table of the requested resource;

comparing the verification signature for the requested resource to the list of resource signatures;~~ and~~

executing the requested resource if the resource signature matches the verification signature and ~~determining~~ the accessibility status is loadable ~~of the requested resource in accordance with the accessibility status by which the resource signature matching the verification signature is classified~~; and

preventing the requested resource from execution if the resource signature matches the verification signature and the accessibility status is restricted.

2. (Original) A method according to Claim 1, wherein the resources include applications or programs.

3. (Currently Amended) A method according to Claim 1, wherein each of the resource signature is generated based further on one or more dynamic link library (DLL) names, and wherein the verification signature is generated based further on one or more dynamic link library (DLL) names ~~the resource signature for each of the respective resources includes a manipulation of structure-related data from the resource~~.

4. (Canceled).

5. (Currently Amended) A method according to Claim 1, wherein generating the verification signature for the requested resource includes:

retrieving a plurality of names ~~data that uniquely identifies the resource~~ from the import table, wherein the plurality of names at least include function name;

sorting the retrieved names ~~data~~;

concatenating ~~linking~~ the sorted names ~~data~~; and

executing a cryptographic ~~mathematical~~ manipulation of the concatenated names ~~linked data~~.

6. (Currently Amended) A method according to Claim 5, wherein executing a cryptographic manipulation includes executing a hash function ~~the resources include applications or programs~~.

7. (Currently Amended) A method according to Claim 5, wherein the plurality of names further includes DLL names ~~retrieved data includes an import table~~.

8. (Currently Amended) A method according to Claim 1, further comprising executing the requested resource if the resource signature does not match the verification signature and the accessibility status is restricted ~~Claim 5~~, ~~wherein the retrieved data includes function names from an import table~~.

9. (Currently Amended) A method according to Claim 1, further comprising preventing the requested resource from execution if the resource signature does not match the verification signature and the accessibility status is loadable ~~Claim 5, the retrieved data includes dynamic link library (DLL) names from an import table~~.

10. (Original) A method according to Claim 1, wherein a same procedure is followed to generate each of the resource signatures and to generate a verification signature.

11. (Currently Amended) A method according to Claim 1, further comprising coding the generated list of resource signatures into a dynamic link library (DLL) of an operating system kernel~~ wherein the accessibility status of the resources includes one of permissible or impermissible~~.

12. (Withdrawn) A method for generating an application identifier, comprising:

receiving a command to generate an identifier for an application;

retrieving an import table from the application;

sorting information from the retrieved import table; and

performing a cryptographic function on the sorted information.

13. (Withdrawn) A method according to Claim 12, wherein the import table is retrieved from an executable of the application.

14. (Withdrawn) A method according to Claim 12, wherein sorting information from the retrieved import table includes sorting function names according to at least one predetermined criterion.

15. (Withdrawn) A method according to Claim 12, wherein performing a cryptographic function on the sorted information includes performing a one-way hash function.

16. (Currently Amended) A method of restricting particular applications, comprising:

receiving a list of application fingerprints corresponding respectively to restricted applications;

receiving a request to execute an application;

generating a confirmation fingerprint for the requested application, wherein the confirmation finger print is generated at least from function names included in an import table of the requested application;

comparing the confirmation fingerprint to the list of application fingerprints; and

restricting the requested application if the confirmation fingerprint matches one of the application fingerprints respectively corresponding to restricted applications.

17. (Currently Amended) A method according to Claim 16, wherein generating a confirmation fingerprint for the requested application includes:

Retrieving a plurality of names from the import table, wherein the plurality of names include function names ~~data from an executable of the requested application describing linkages to other applications~~;

sorting the retrieved names ~~data~~;

concatenating ~~organizing~~ the sorted names ~~information~~ in a predetermined manner; and

hashing the organized names ~~information~~.

18. (Currently Amended) A method according to Claim 17, wherein the confirmation fingerprint is further generated from DLL names included in the import table of the requested application, and wherein retrieving a plurality of

names further includes retrieving DLL names ~~the retrieved data includes an import table~~.

19. (Currently Amended) A method according to Claim 17, wherein hashing the organized names include performing one of a Message Digest (MD) 5 hash and a Secure Hash Algorithm (SHA) 1 ~~the sorted information includes function names~~.

20. (Original) A method according to Claim 16, wherein the restricted applications are not licensed.

21. (Withdrawn) An apparatus, comprising:

a licensing manager component to provide identification for an application and to further assign a classification of the application as being restricted or unrestricted; and

a developer component to code an operating system to include the identification in correspondence with the classification.

22. (Withdrawn) An apparatus according to Claim 21, wherein the licensing manager component is to generate identification for an application using an import table from the application.

23. (Withdrawn) An apparatus according to Claim 21, wherein the licensing manager component is to provide a restricted classification for an unlicensed application.

24. (Withdrawn) An apparatus according to Claim 23, wherein, for an unlicensed application, the licensing manager component is to:

retrieve an import table from an executable of the application;

sort and string together function identifiers from the import table; and

execute a cryptographic algorithm on the function identifiers.

25. (Withdrawn) An apparatus according to Claim 24, wherein the cryptographic algorithm is a hashing algorithm.

26. (Currently Amended) An apparatus, comprising:

an interface to receive a request for a running state of an application;

an application identifier to generate an application digital signature ~~an identifier~~ for the application;

an application manager to match the application digital signature ~~identifier~~ against a list of stored digital signatures ~~identifiers~~ indicating whether corresponding applications are eligible or ineligible for a running state; and

an enabler to enable the running state for the application if the application digital signature ~~identifier~~ is not matched to a stored digital signature ~~an identifier~~ indicating that the application is ineligible.

27. (Currently Amended) An apparatus according to Claim 26, wherein the application identifier is to generate an application digital signature ~~identifier~~ using an import table from the application.

28. (Currently Amended) An apparatus according to Claim 27, wherein the application identifier is to:

retrieve an import table from an executable of the application;

sort and string together the function names ~~identifiers~~ from the import table; and

hash the stringed function names ~~identifiers~~.

29. (Currently Amended) An apparatus according to Claim 28, wherein the instruction to hash further includes an ~~and~~ instruction to execute an MD5 hashing algorithm on the stringed function names.

30. (Withdrawn) A computer-accessible medium having one or more instructions that are executable by one or more processors, the one or more instructions causing the one or more processors to:

receive a command to generate a program fingerprint;

sort static data from within the program; and

create a program fingerprint using the sorted static data.

31. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the static data includes an import table.

32. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the static data includes function names corresponding to an import table from an executable of the program.

33. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the one or more instructions that cause the one or more processors to sort static data further cause the one or more processors to:

organize the function names from an import table corresponding to a program executable in accordance with at least one predetermined criterion; and

link the organized function names.

34. (Withdrawn) A computer-accessible medium according to Claim 33, wherein the one or more instructions that cause the one or more processors to create a program fingerprint further cause the one or more processors to execute a hashing algorithm on the sorted static data.

35. (Currently Amended) A computer-accessible storage medium having an application programming interface (API), the API having one or more instructions to cause one or more processors to:

receive a request to run a program;

generate a digital signature for the program;

compare the generated digital signature against a compilation of digital signatures corresponding to restricted programs; and

enable only those programs for which the signature does not match with any of the compiled signatures.

36. (Currently Amended) A computer-accessible storage medium according to Claim 35, wherein the one or more instructions to generate a digital signature for the program cause the one or more processors to sort a list of elements from an import table corresponding to an executable of the program.

37. (Currently Amended) A computer-accessible storage medium according to Claim 36, wherein the one or more instructions to generate a digital signature

for the program cause the one or more processors to hash a sorted list of function names from the import table.

38. (Currently Amended) A computer-~~executable-~~accessible storage medium according to Claim 35, wherein the API is included in an operating system.

39. (Currently Amended) A computer-~~executable-~~accessible storage medium according to Claim 35, wherein the operating system runs on a web server.

40. (Currently Amended) A computer--~~executable-~~accessible storage medium according to Claim 35, wherein the operating system runs on an application server.

41. (Currently Amended) A license enforcement method, comprising:

generating a digital signature for each of a plurality of applications;

classifying each of the digital signatures in accordance with a licensing status for the corresponding applications;

coding an operating system to:

include the classified digital signatures,

generate a digital signature for a requested application,

compare~~-map~~ the digital signature for the requested application to the classified digital signatures, and

run the requested application when the digital signature for the requested application does not map to digital signature classified as not being licensed.

42. (Original) A license enforcement method according to Claim 41, further comprising downloading an updated list of the classified digital signatures to the operating system.

43. (New) A method according to Claim 7, wherein executing a cryptographic manipulation includes executing a hash function.